LESSONS LEARNED IN THE DEVELOPMENT OF THE STOL INTELLIGENT TUTORING SYSTEM

Thomas Seamster, Ph.D
Mr. Clifford Baker
Carlow Associates Incorporated

Mr. Troy Ames
NASA Goddard Space Flight Center

ABSTRACT

This paper presents lessons learned during the development of NASA's Systems Test and Operations Language (STOL) Intelligent Tutoring System (ITS), being developed at NASA Goddard Space Flight Center with support by Carlow Associates Incorporated and Computer Sciences Corporation. The purpose of the intelligent tutor is to train STOL users by adapting tutoring based on inferred student strengths and weaknesses. This system has been under development for over one year and numerous lessons learned have emerged. These observations are presented in three sections, as follows. The first section addresses the methodology employed in the development of the STOL ITS and briefly presents the ITS architecture. The second presents lessons learned, in the areas of:

- intelligent tutor development
- documentation and reporting
- cost and schedule control
- tools and shells effectiveness

The third section presents recommendations which may be considered by other ITS developers, addressing: access, use and selection of subject matter experts; steps involved in ITS development; use of ITS interface design prototypes as part of knowledge engineering, and; tools and shells effectiveness.

BACKGROUND

This paper presents lessons learned during development of the STOL ITS, which is being developed to assist NASA control center personnel in learning STOL.

The STOL ITS is currently being designed to provide the Gamma Ray Observatory (GRO) Flight Operations Team (FOT) with introductory and refresher training/tutoring on STOL and its main applications. The initial orientation of this effort was to provide intelligent tutoring in the context of complex dynamic systems. This initial effort was concerned with the application of ITS technology to improve system performance and safety in supervisory control. The emphasis was on the modeling of operator's intentions in the form of goals, plans, tasks, and actions. More recently, this research has expanded to include a review of ITS technology as it may be applied to the tutoring of command and control languages (Eike, Seamster, & Truszkowski, 1989).

A related effort is taking place at NASA Johnson Space Center (JSC) through the development of PD/ICAT, an ITS to train Mission Control Center Flight Dynamics Offices to deploy a specific type of satellite for the Space Shuttle (Loftin, Wang, Baffes, & Hua, 1988). PD/ICAT is part of a larger effort to develop a general architecture for intelligent tutoring/training systems to train controllers in the performance of mission-critical tasks. This architecture consists of a blackboard that serves as a communications interface for the following system's components: user interface, expert module, trainee model, training scenario generator, and a training session manager. The developing STOL ITS design utilizes the NASA/JSC general architecture with several modifications that will allow for expanded capability and additional flexibility (Seamster, 1990).

Purpose

The purpose of the STOL ITS is to facilitate understanding, development and application of STOL directives and procedures. Since this project is primarily a research effort (as opposed to a development effort), several research questions are being addressed, including:

1. the feasibility of developing a general purpose STOL tutor which can be effectively utilized in a wide range of POCC environments;
2. evaluation of the relative effectiveness of alternative knowledge representat-ions for tutoring a command and control language;
3. evaluation of the relative effectiveness of alternative user interface modes for students with varying skill levels.

A major element of this project is to demonstrate the effectiveness of ITS technology in a NASA control room environment by selecting a set of representative STOL tasks which have application across a wide range of POCCs.

Project Phases

STOL ITS development is proceeding in three main phases, as follows.

Feasibility Stage - The focus of this stage involves evaluating the feasibility of developing a general command language ITS. Two aspects to feasibility have been addressed: technical and cost. Technical feasibility refers to reasonableness of developing an ITS to tutor STOL. In this context, feasibility is determined by the availability of Domain Experts (DE's) capable of expressing the nature of their expertise in a form which can be translated into an expert system.

Cost feasibility refers to the probability that the costs incurred in developing the ITS can be recovered through improvements in POCC STOL training. In order to justify the development costs associated with an ITS, there must be a relatively high number of students processed through the system. Since no individual POCC is likely to have a sufficiently large number of students to justify the cost, the ITS must be applicable across several operating environments. In order to meet this objective, the operational version of the STOL ITS is being designed to be portable and modular, emphasizing a decomposition where specific command language functions are separated from the general command language tutoring functions.

Research/Extensibility Stage - The object-ive of this stage is to expand the rule set to encompass a relatively complete set of DE cognitive functions. One of the main objectives of this stage has been to develop the various ITS modules to the point that they can interact (i.e., exchange data) to perform the functions of an intelligent tutoring system. Consistent with the objective to reduce costs, these modules operate under a common operating system, and, where possible, are developed with existing NASA data, rule sets and development tools. The general ITS architecture developed at NASA/JSC (Loftin et al., 1988) has been used. A parallel effort has been to develop and validate the user interface for the ITS. This effort consists of an iterative series of develop-ment / demonstration / revision cycles, in which potential users of the ITS evaluate and comment on the user interface. The focus of this effort has been directed at investigating the relative efficacy of alternative forms of problem/knowledge representation for students with varying levels of experience and expertise.

Field/Demonstration Stage - The objective of this phase is to demonstrate/evaluate the viability of the STOL ITS. As part of field demonstration of the GRO STOL ITS, the STOL ITS provides the trainee with several aids to facilitate learning. The first aid is an alternative representation of STOL which is graphically oriented, and which consists of dynamic icons and graphics representing the various elements of the GRO environment. This representation is used to facilitate learning STOL and is treated as an intermediate step between the trainee's current way of thinking about the system and the way he or she will use STOL in the

control room. Another aid is a hypertext glossary of STOL directives, containing semantics, syntax and examples. Finally, a STOL Certification tool has been developed which measures a students STOL/GRO proficiency and provides practice in use of the STOL language as applied to GRO.

Knowledge Acquisition Process

The STOL ITS is designed primarily for the mission analysts of the GRO FOT. There are to be 9 individuals serving in this capacity. In addition, the STOL ITS, in its initial version, may be used by spacecraft engineers and guest scientists. With some modification, the ITS may also be used by Multi-Satellite Operations Control Center (MSOCC) operators, and by analysts on other missions. As was mentioned in the introduction, STOL ITS is being developed to ultimately serve a number of different missions, and a number of its modules may be used as the basis for tutors of other NASA command and control languages.

The GRO FOT mission analysts include the Operations Controllers (OCs) and the Command Operators (COs). The OC is the senior person who directs the activities in the GRO Missions Operation Room (MOR), and the CO executes the specific STOL commands. The OC is the senior person who directs the activities in the MOR, and the CO executes the specific STOL directives. In addition to these two operators, there is a mission planner present during the day shift and a group of engineers who will be in the GRO MOR when required. This paper concentrates on the tasks of the two primary operators, the OC and CO. The OC evaluates the status of GRO related systems and subsystems. The OC uses at least three CRTs to monitor the different types of status data. The OC also interacts with a number of people in the execution of his job. In addition to the CO, the OC interacts with other network operators via a voice link communication, making the communication system an important part of the OC job. Given this background, the ten steps of the STOL ITS Knowledge Acquisition Process are presented in Table 1.

Table 1. Ten Steps in the STOL ITS Knowledge Acquisition Process

1. Extract STOL and MSOCC terms from documentation
2. Sort and cluster analysis of STOL elements
3. Identify key directives and SMEs for the project
4. Identify the key concepts and potential tutor problems
5. Rate and select key problem/solutions for the demonstration tutor
6. Develop problem representations
7. Gather SME protocols for set of problems and transcribing
8. Analyze first set of protocols and developing pseudo-code rules
9. Expand and refine first set of pseudo-code rules and provide novice misconceptions
10. Translate rules into CLIPS

A collateral step to the knowledge engineering (KE) process was the development of the STOL ITS user interface. This interface, developed in HyperCard, controls the presentation of information to the STOL ITS students, and interfaces with the CLIPS module which house the ES portions of the ITS.

STOL ITS Tools and Interfaces

STOL Certification Tool. The purpose of this tool was threefold. First, it was developed to refine and verify STOL user error classifications, such as error tendencies of students as a function of STOL experience and spaced based platform experience. Secondly, it was developed to measure the progress of students as they used the STOL ITS. Finally, it has been used as method to

extensively practice issuance of STOL commands to GRO, with feedback provided to students as to the efficiency of their command operations. The Certification tool:

- collects error data on STOL users (commands, arguments, syntax)
- provides a basis for developing and validating STOL ITS student models
- provides a tool for certification of STOL users after and during training (including STOL ITS training)
- practices student issuance of STOL commands to GRO

The certification tool has a simple interface and architecture. Four sections are provided for student testing/practicing of STOL commands for GRO major mission sections. These are: Introductory; Prepass; On Pass, and; Post Pass.

Within these sections, there are approximately 200 questions which can be posed to students, with one to three answers being considered correct for each question. In the course of a session, the Certification toll poses a problem, and the student responds, after which the tool determines whether the response is correct, logs the data, and poses a subsequent problem. Figure 1 shows a certification tool screen with a student response.

Hypertext STOL Glossary. The student may also access a detailed Hypertext STOL glossary, which presents:

- STOL directives
- Aliases
- Required and optional arguments for directives
- Descriptive information related to directives
- Modal use of directives

The glossary is used in the course of a session when a student has difficulty in responding to a posed question. Figure 2 presents the basic interface for the STOL Glossary.

STOL ITS Interface. Development of the STOL ITS has placed considerable emphasis the design of the interface. This has been for the following reasons:

- ITS development matured to the point of becoming user-centered
- complex, relational information to be presented
- the use of the interface to gather student data
- the use of the interface to evaluate different tutoring strategies
- the use of the interface in knowledge acquisition
- The interface provides a point of convergence for the various STOL ITS modules

Figures 3 and 4 present example of STOL ITS interface screens.

LESSONS LEARNED

Lessons learned in the development of this ITS are related to: intelligent tutor development in general; documentation and reporting; Cost and schedule control, and; Tools and shells effectiveness.

General Lessons

Subject Matter Expert Access. It was estimated that the development of the STOL ITS would involve a number of interactions with the user community. This community, the GRO FOT, consists of 20 individuals who would serve as Subject Matter Experts (SMEs) and novice users. The SMEs were to serve as the main source of STOL information during the knowledge acquisition phase. Novices would be used to evaluate the prototype user interface as well as the prototype STOL ITS. A total of 200 hours was requested from the GRO FOT. The GRO FOT's initial reaction was that it would be difficult to provide that many hours. Part of the difficulty was that the GRO FOT would be in the middle of End-to-End (ETE) tests in preparation for launch. It became evident that if the GRO FOT had to provide 200 hours, it would not be able to participate in the STOL ITS development process. Because of this, the SME and novice trainee needs were reevaluated, and a

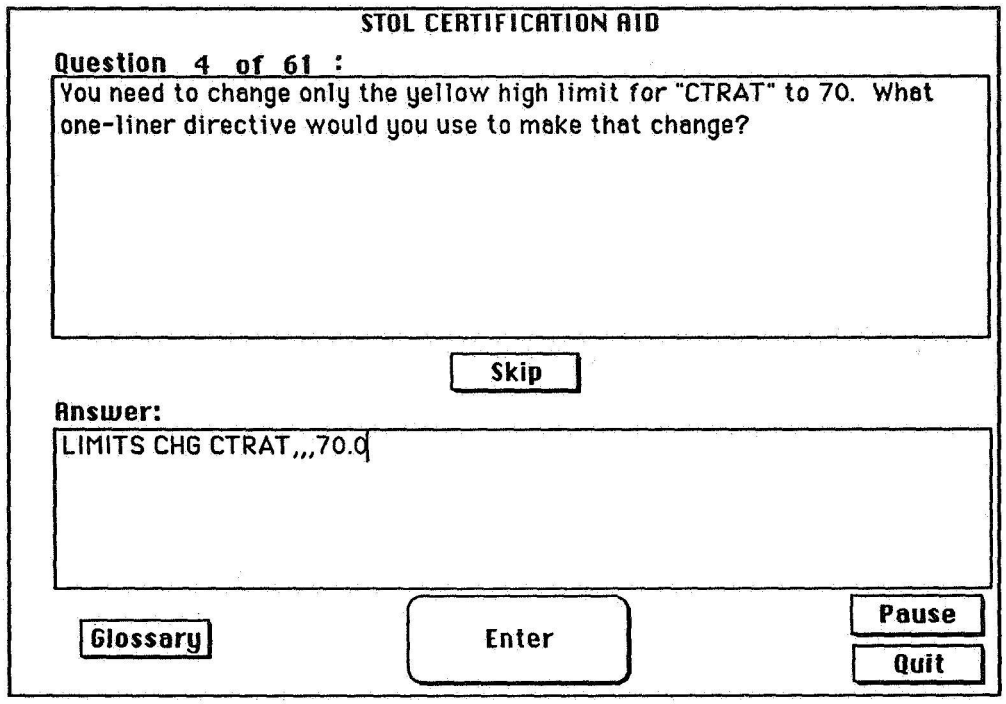

Figure 1.
Certification Tool Interface with Student Response

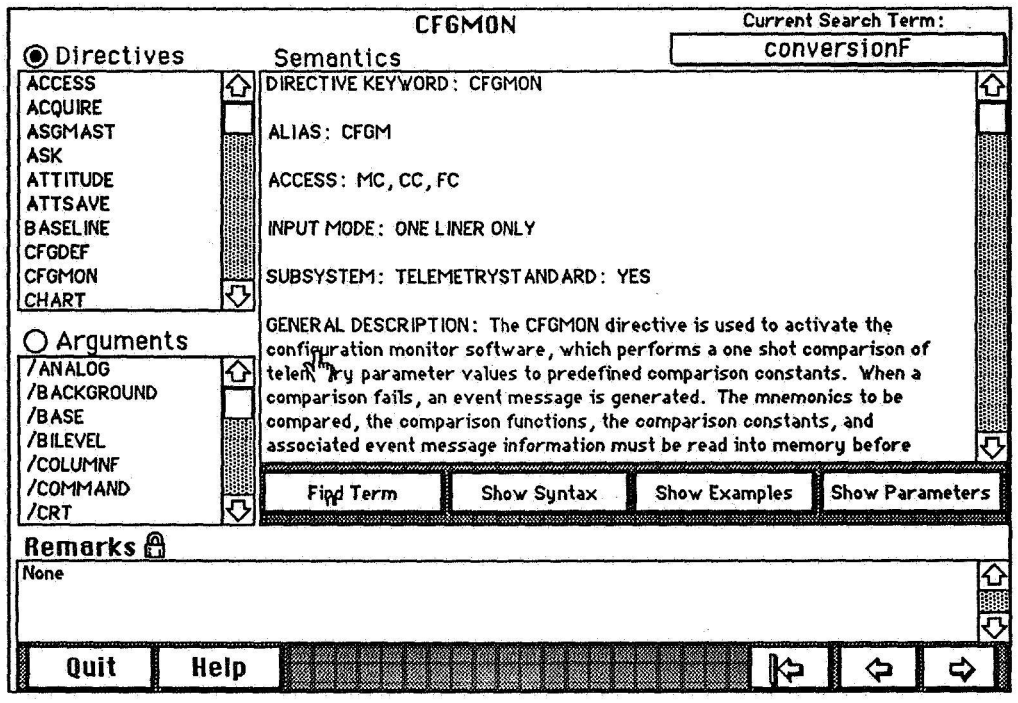

Figure 2
Basic Interface of the STOL Glossary

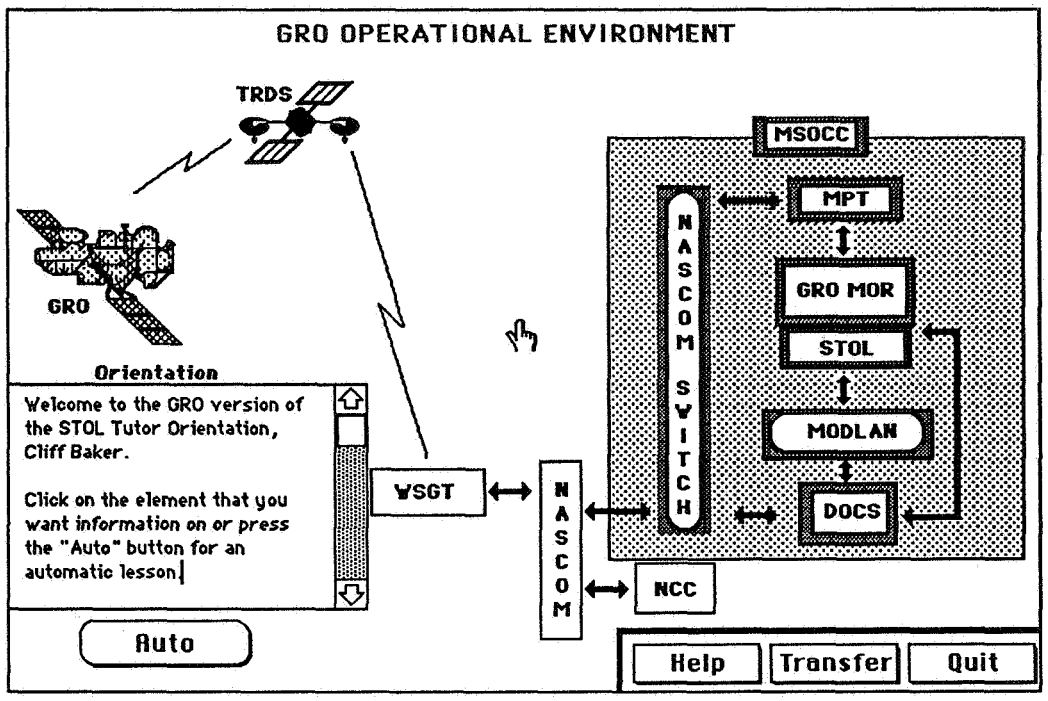

Figure 3
GRO Orientation Screen of the STOL ITS

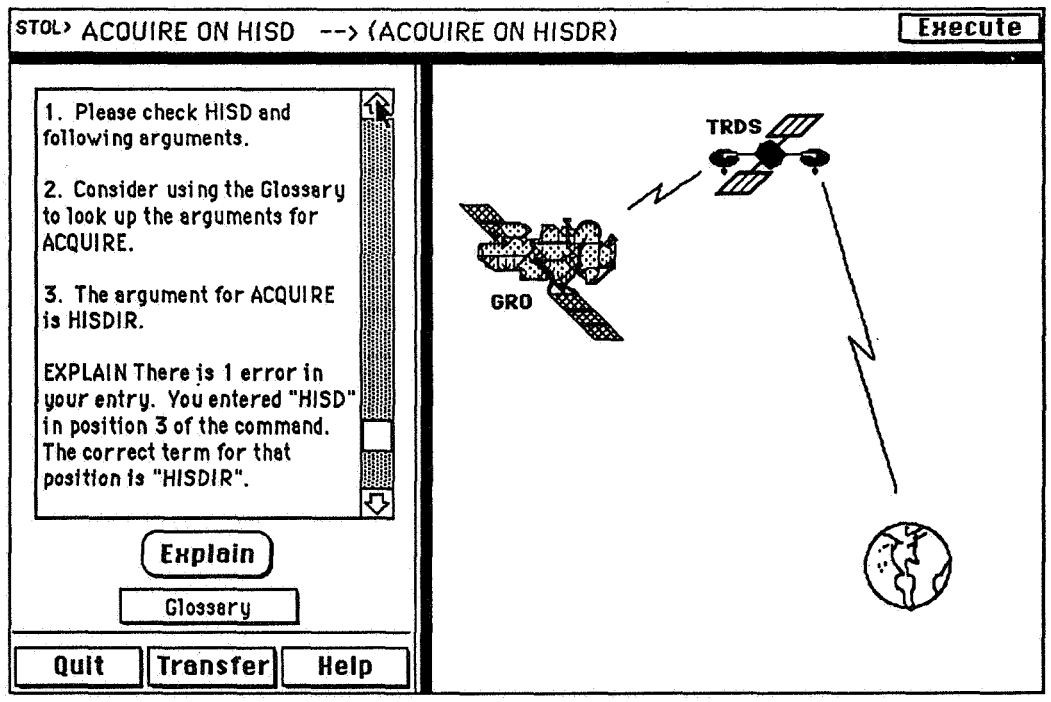

Figure 4
Advanced Prepass Test/Tutoring Screen of the STOL ITS

new approach was developed in order that the tutor could be developed with a reduction in GRO FOT time. By providing the GRO FOT with a Glossary and Certification Tool, addition contact time was provided outside the formal structure of the knowledge engineering process.

Based on the experience of STOL ITS development, particular missions may be most willing to provide expert hours if they benefit directly from the ITS. The experts for a particular mission are in great demand, and it can be difficult to justify their time on non-mission related projects. If the ITS project can directly benefit mission training, then the FOT may be more likely to get involved.

FOT/SME Training Cycle Coincidence with ITS Development Cycle. There is a general mismatch between the development cycle of an ITS and the training cycle of an FOT. In addition to the actual number of hours required of a mission, it has proved important to consider when those hours are needed, and how the entire ITS development process will fit into the FOT's training needs. A persistent problem was the KE's need for specific types of experts or novices which do not coincide with their availability within the FOT. At the beginning of the FOT training cycle, there are usually several analysts who are spending up to six months gaining expertise on the specific spacecraft, but who are not sufficiently knowledgeable to serve as SMEs. The lesson here is that, for a new ITS, sufficient domain expert knowledge may not be available to develop an ITS which meets or precedes the needs of the FOT training cycle.

When it became evident that the STOL ITS would not be delivered until after the FOT had been certified, the STOL ITS development team decided to meet the needs of the GRO FOT by providing them with a certification tool that could be used in the normal FOT training cycle. That certification tool provides the STOL ITS development team with substantial information on trainee errors. In general, the ITS development teams needs to take a flexible and opportunistic approach to the development process.

Phase Activities. In developing an expert system, one approach, such as that recommended by the Expert System Development Methodology (ESDM), is to start with a key concept or cognitive function and develop a small system to evaluate the feasibility of the system. This is a useful first stage for some systems, but in the case of ITS development, it proved more efficient and compelling to first develop a prototype of the user interface. The prototype user interface is a much less abstract representation of the final system when compared with a small rule set that models some expertise. The prototype user interface can clearly demonstrate to users the intended functionality of the system and can provide early feedback on the proposed tutor capabilities.

Project Reporting Lessons

The STOL ITS development project has generated a subset of the Expert System Development Methodology (ESDM) reporting requirements (CSC, 1989). ESDM structures expert systems development as an iterating process made up of the five stages shown in Table 2.

In general, ESDM has provided a good framework for the development of expert systems including ITSs. The STOL ITS development process spanned only three of the ESDM stages, and as such could not provide a complete evaluation of the ESDM process. This limited evaluation of ESDM has pointed out a number of issues. ESDM has been selectively employed for about nine months of the STOL ITS development cycle including most of the research prototype and a partial field prototype development. If the ESDM process had been fully implemented, approximately 10 to 12 reports would have been prepared. A significant amount of the development time is required to generate reports. One approach is to take the following ESDM recommended reports and consolidate them into two reports per stage preceded by a project initiation report and terminated with a final project report. The reports shown in bold were the most useful for the STOL ITS development cycle.

Table 2. ESDM Stages and the Equivalent STOL ITS Stages

ESDM Stages
Feasibility prototype
Research prototype
Field prototype
Production prototype
Operational prototype
STOL ITS Development Cycle
User interface development
Research prototype
Partial field prototype

(Asterisked reports are due for every stage of ESDM development):

- Concept and Project Initiation Report
- **Stage Project Management Plan***
- Prototype Design Report*
- Prototype Operations Guide*
- **Prototype T & E Report***
- **Knowledge Engineering Report***
- **Technology Transfer Report**
- Project Termination Report

In conclusion, ESDM provided a valuable framework, and its modification and partial automation would provide an extremely valuable tool for the management of KE processes.

Project Cost and Schedule Control Effectiveness Lessons

Use of an Existing ITS Architectures, Tools, and Prototypers. One of the objectives of the STOL ITS project was to promote ways to reduce the development costs of the ITS. Use of an existing ITS architecture helped to greatly reduce the time of the STOL ITS development effort. The development costs for the STOL ITS were reduced by using NASA/JSC's general architecture. The use of that architecture reduced the ITS control coding requirements. Next, the user interface was prototyped in HyperCard which provided substantially reduced development times when compared with coding in C or some other programming language. Finally, CLIPS provided a more efficient expert system development environment than AI languages such as LISP or Prolog.

Figure 5 shows the estimated development type for the STOL ITS prototype to be about 1.5 man years. It was estimated that it would take about one-half a man year for each of the following three tasks: 1) Develop a prototype user interface, 2) Develop the CLIPS rules, and 3) Integrate the user interface with the rules .

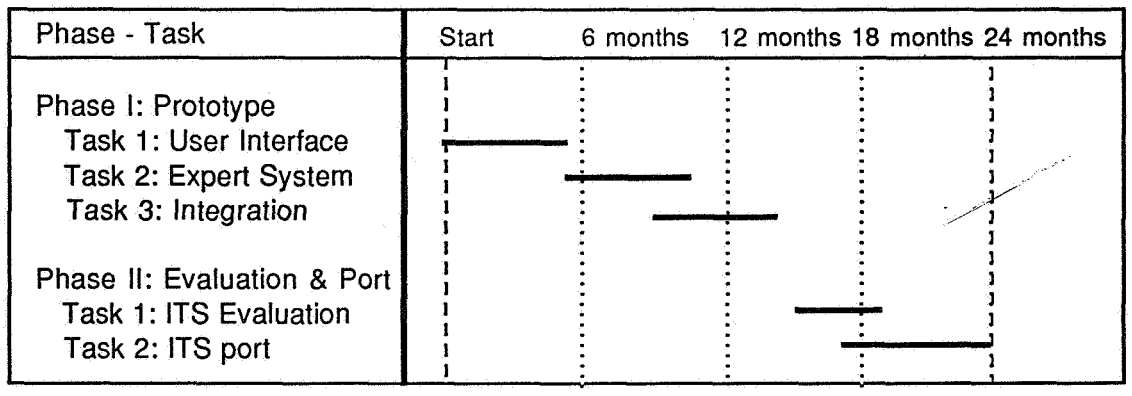

Figure 5. Estimated STOL ITS Schedule in Man-Months (Seamster, 1989)

The actual development time for the STOL ITS demonstration prototype came close to that, and is substantially shorter than the estimated 6 man years that it took to develop an operational version of NASA Johnson Space Center's Payload-assist module Deploys/Intelligent Computer Aided Training (PD/ICAT). There are some difficulties in comparing estimates for developing a prototype versus estimates for an operational system, but it is evident that the project costs were greatly reduced through the used of a general architecture and development tools. An even greater savings could be realized when using STOL ITS's modules and the general ITS prototyper in developing other ITSs for NASA command and control languages.

Tools & Shells Effectiveness Lessons

The STOL ITS development process has brought together a number of existing tools and architectures to form a general ITS prototyper. This ITS prototyper has allowed for the relatively rapid development of an ITS. The general ITS prototyper is based on the following tools and applications: HyperCard, CLIPS, HyperCLIPS, and PD/ICAT. With some modification, the domain independent PD/ICAT rules were used as the basis of the control of the ITS prototyper.

General Need for Knowledge Acquisition Tools. During the KE process, a number of different software applications were utilized to perform the required data analyses. For example, a spreadsheet program was used to build the proximity matrices, and a statistical analysis program was used to run the cluster analysis. This process resulted in a large number of steps and substantial time spent entering and verifying data. As has been noted by other knowledge engineers, there is a great need for knowledge acquisition tools that can simplify the data collection and data analysis process.

In the early phases of the STOL ITS development, a brief review was made of existing knowledge acquisition tools with the conclusion that they are either too specialized or do not provide sufficient flexibility so that their results can be easily imported to other tools. This points out the strong need for a flexible and integrated set of knowledge engineering tools that can be used for collecting, analyzing, and reporting KE results.

HyperCard V.1.2.5. HyperCard, a Macintosh prototyping tool, was used to develop STOL ITS interfaces. It allowed for the rapid development of user interface prototypes by facilitating the design of buttons, dialogs, and interactive graphics, and it allowed for quick evaluation and modification of interfaces after review by potential STOL ITS end users. The major limitations of the 1.2.5 and earlier versions of HyperCard (screen surface size, graphics limitations) have been overcome by version 2.0 of Hypercard. SuperCard also provides these capabilities and may be a more flexible tool for ITS user interface prototyping.

CLIPS V 4.3 and HyperCLIPS V 1.0.2. The C Language Integrated Production System (CLIPS) was developed at the Johnson Space Center by the Artificial Intelligence Section for use in developing systems for the NASA Mission Control Center. CLIPS uses a forward chaining rule system with a syntax allowing free form patterns as well as single and multi-field variable bindings across patterns. CLIPS, implemented in the C language, is highly portable. The primary method of representing knowledge in CLIPS is a rule, a collection of conditions and the actions to be taken if the conditions are met. The expert system developer defines the rules which describe how to solve a problem, and the entire rule set makes up the knowledge base. CLIPS provides the inference engine which matches the rules to the current state of the system and applies the actions or consequents.

HyperCLIPS permits the execution of CLIPS rules from HyperCard. By integrating HyperCard and CLIPS rapid prototyping of knowledge-based expert systems may be accomplished. HyperCLIPS includes a number of programs which implements a specialized version of an XCMD. The CLIPS XCMD is named "ClipsX" and is

designed to be as similar to the command line version of CLIPS. For this version of HyperCLIPS, the commands implemented are: clear, load, reset, and run. Data returned from CLIPS may be retrieved through the HyperTalk function, *the result*, after calling the run command. HyperCLIPS proved very useful in integrating the STOL ITS HyperCard user interface with the CLIPS rules. The primary limitation is the reduced number of CLIPS commands that are implemented. For example, it would be very useful if the CLIPS command unwatch all, could be passed prior to loading the CLIPS rules. This would save a substantial amount of time during the loading process. For prototyping purposes, HyperCLIPS V 1.0.2 has been very effective.

PD/ICAT. The STOL ITS architecture is based on a subset of the PD/ICAT rules which were subsequently modified into a general ITS prototyper. PD/ICAT was developed for NASA/JSC (Loftin, Wang, Baffes, & Hua; 1988), and was designed around a general ITS architecture that could be used for a number of different procedural training tasks. Based on the analysis of the PD/ICAT modules, the STOL ITS project utilized the rules from ERROR.CLP, LINKRULE.CLP, MANAGER.CLP, and SUPPORT.CLP as the primary control rules for the CLIPS portion of the general prototyping ITS architecture. These four modules were selected because they are relatively domain independent, and could provide the basic control for the ITS prototyper. The major form of modification of the PD/ICAT rules has been the replacement of the external functions that handle the output of the rules back to the user interface. The use of the PD/ICAT general architecture reduced the STOL ITS development time substantially and inspired the idea to develop a general ITS prototyper.

CONCLUSIONS AND RECOMMENDATIONS

Subject Matter Expert Access. Missions may provide SMEs to support the KE process if they benefit directly from ITS development. The experts for a particular mission are in great demand, and it can be difficult to justify their time on non-mission related projects. Also, provision of development tools (Such as the Certification Tool) can increase SME contact time, and provide immediate support to the mission training program.

Prototyping. It proved efficient to develop early on a prototype of the user interface. This provides at least two strong advantages. First, the resulting prototype can be used to demonstrate the ITS to potential users and gain SME support and feedback. Secondly, it can be used in the early stages to more clearly define the requirements and capabilities of the system. Starting with a prototype of the ITS is a user-centered approach which can ensure that the final system will meet the needs and requirements of the trainees.

Documentation and Reporting. By consolidating ESDM reports, duplication of effort can be reduced, and the time spent preparing reports can also be reduced. For example, the management plan and design report could be combined to form an initial report for each stage. The operations guide and test and evaluation could be combined for a stage termination report. The knowledge engineering report proved to be a very useful report, but it is not clear that it is needed at every stage.

Cost and schedule control. Project costs were greatly reduced through the used of a general architecture and development tools. Greater savings could be realized when using STOL ITS's modules and a general ITS prototyper in developing other ITSs for NASA command and control languages.

Tools and shells effectiveness. A major part of the development of most expert systems, including Intelligent Tutoring Systems (ITS), is the prototyping phase. The development of knowledge-based systems requires prototyping for most of the stages, and consequently, the prototyping environment is critical to the success of such efforts. Prototyping has proven central to the development of the STOL ITS, and in the same way that STOL ITS benefited from NASA/JSC PD-ICAT development, future NASA ITS development could benefit from

the STOL ITS KE process and tools. A general ITS prototyper could serve as a foundation for the development of future ITSs.

During the current STOL ITS effort, a number of general ITS prototyper requirements have been identified. That process could be expanded by the preparation of a formal requirements analysis for a NASA general ITS prototyper. These requirements would be extended to meet the needs of the NASA ITS community emphasizing the development of a prototyper which would allow for the quick development of ITS prototypes in the microcomputer environment. The new requirements would be analyzed to identify modifications and additions that need to be made to the STOL ITS prototyper. These modifications could then be implemented resulting in a general ITS prototyper to serve the ITS development needs of NASA.

GRO FOT STOL ITS Refinement. The current structure of the trainee model is based on that used in PD/ICAT, but the STOL ITS would be greatly improved if it were to adopt a more sophisticated model. The current trainee model has been developed as a database containing information about the trainee's understanding of specific productions. The STOL ITS tutoring relies on providing the trainee with practice of specific productions, and the trainee model keeps track of the productions' status. As the trainee makes mistakes with underlying productions, the trainee model database keeps track of those productions, and that data base can anticipate when the trainee might make an error on an advanced problem containing that production.

The STOL ITS demonstration prototype would make a good testbed for hypermedia application. The current version of STOL ITS is limited to the display of still pictures and text, and could be expanded to include voice, moving video images, and other interactive capabilities of hypermedia. For example, the STOL ITS Orientation Lesson shows all major elements of the GRO control system. In its current implementation, when the trainee selects an object, a static picture and/or text is presented to give additional detail. This type of tutoring would be enhanced if hypermedia were available to present a more realistic and animated view of the GRO operational environment.

REFERENCES

Anderson, J. R. (1987). Production systems, learning, and tutoring. In D. Klahr, P. Langley, and R. Neches (Eds.), Production system models of learning and development (pp. 437-458). Cambridge, MIT Press.

Anderson, J. R., Conrad, F. G., & Corbett, A. T. (1989). Skill acquisition and the LISP tutor. Cognitive Science, 13, 467-505.

CSC (1989). Expert System Development Methodology User Guide. Prepared for NASA Goddard Space Flight Center, August 1989.

Eike, D. R., Seamster, T. L., & Baker, C. C. (1989). Applying Intelligent Tutoring System Technology to NASA Control Centers. (Draft) White Paper Report prepared for NASA Goddard Space Flight Center, May, 1989.

Eike, D. R., Seamster, T. L., & Truszkowski, W. (1989). Functional description of a command and control language tutor. Proceedings of SOAR'89: Third Annual Workshop on Space Operations Automation and Robotics (585-592). Houston, Texas.

Loftin, R. B., Wang, L., Baffes, P., & Hua, G. (1988). An intelligent training system for Space Shuttle flight controllers. Proceedings of the 1988 Goddard Conference on Space Applications for Artificial Intelligence (pp. 3-15). Greenbelt, MD.

Mitchell, C. M. (1989). Human-Computer Interaction in Distributed Supervisory Control. Semi-Annual Report prepared for NASA Goddard Space Flight Center, August, 1989.

Seamster, T. L. (1989). Cognitive Task Analysis: Techniques for the Development of Airborne Weapons Training. Final Report prepared for the Data Systems Engineering, Oak Ridge National Laboratory.

Seamster, T. L. (1989). Functional Requirements Document: Systems Test and Operations Language (STOL) Intelligent Tutoring System (ITS). (Draft) Technical Report prepared for NASA Goddard Space Flight Center, August, 1989.

Seamster, T. L. (1990). Architecture Review: General Intelligent Tutoring System (ITS) Prototyper. Draft Report prepared for NASA Goddard Space Flight Center, April, 1990.

Seamster, T. L. (1990). Systems Test and Operations Language (STOL) Intelligent Tutoring Systems (ITS) Knowledge Engineering Report. (Draft) Technical Report prepared for NASA Goddard Space Flight Center, July, 1990.

Seamster, T. L. (1990). Architecture Review: General Intelligent Tutoring System (ITS) Prototyper. (Draft) Technical Report prepared for NASA Goddard Space Flight Center, April, 1990.